

PLP - Recuperatorio del Primer Parcial - 1^{er} cuatrimestre de 2022

Este examen se aprueba obteniendo al menos dos ejercicios bien menos (B-) y uno regular (R). Las notas para cada ejercicio son: -, I, R, B-, B. Entregar cada ejercicio en hojas separadas. Poner nombre, apellido, número de orden y cantidad de hojas en la primera hoja, y numerar las hojas. Se puede utilizar todo lo definido en las prácticas y todo lo que se dio en clase, colocando referencias claras.

El orden de los ejercicios es arbitrario. Recomendamos leer el parcial completo antes de empezar a resolver.

Ejercicio 1 - Programación funcional En este ejercicio no se permite el uso de recursión explícita, a menos que se indique lo contrario.

Se desea modelar en Haskell los árboles con información en las hojas (y sólo en ellas). Para esto introduciremos el siguiente tipo:

```
data AIH a = Hoja a | Bin (AIH a) (AIH a)
```

Se pide:

- Definir el esquema de recursión estructural `foldAIH` y dar su tipo. Por tratarse del primer esquema de recursión que tenemos para este tipo, se permite usar recursión explícita.
- Escribir las funciones `altura :: AIH a -> Integer` y `tamaño :: AIH a -> Integer`. Considerar que la altura de una hoja es 1 y el tamaño de un AIH es su cantidad de hojas.
- Definir la lista (infinita) de todos los AIH cuyas hojas tienen tipo `()`¹. Se recomienda definir una función auxiliar. Para este ejercicio se permite utilizar recursión explícita.
- Explicar por qué la recursión utilizada en el punto c) no es estructural.

Ejercicio 2 - Cálculo lambda

Se desea extender el cálculo lambda con el tipo $BDD_{\sigma, \tau}$, que representa un diagrama de decisiones binarias que, al evaluarse en valores de tipo σ , arrojará resultados de tipo τ . Es decir, un árbol binario no vacío cuyos nodos internos contienen predicados (funciones que devuelven valores de tipo `Bool`), los cuales se podrán evaluar en un valor dado para definir por qué rama descender hasta llegar al resultado final. ($BDD = Binary Decision Diagram$).

$\sigma ::= \dots \mid BDD_{\sigma, \tau} \quad M ::= val_{\sigma}(M) \mid M ? M ; M \mid evaluar(M, M)$

donde:

- $BDD_{\sigma, \tau}$ es el tipo de los diagramas de decisiones binarias que admiten valores de tipo σ y arrojan resultados de tipo τ .
- $val_{\sigma}(M)$ representa un diagrama sin predicados, que al evaluarse en un valor de tipo σ siempre arroja como resultado M - o, más precisamente, el resultado de reducir M a un valor.
- $F ? M ; N$ representa al diagrama cuya raíz es la función F , y sus subárboles son M y N . Si al evaluar el diagrama para un valor V la aplicación de F a V devuelve `True`, se continuará evaluando el subárbol M . En caso contrario, se descenderá por el subárbol N .
- $evaluar(M, N)$ representa la evaluación del diagrama M para el argumento N . Para obtener el valor de esta evaluación se deberá aplicar, en cada paso, la función del nodo actual al valor de N , para luego descender por la rama que corresponda según el resultado de la aplicación sea `True` o `False`.

Se pide:

- Dar las reglas de tipado para soportar los nuevos términos.

¹El tipo `()`, usualmente conocido como *unit*, tiene un único valor, denotado como `()`.

- b) Describir el nuevo conjunto de valores y dar las reglas de reducción en un paso para los nuevos términos.
- c) Mostrar paso a paso cómo reduce el siguiente término:
- evaluar(M , if True then 0 else Succ(0))

donde $M \stackrel{\text{def}}{=} (\lambda x: \text{Nat}. \text{True}) ? ((\lambda y: \text{Nat}. \text{isZero}(y)) ? \text{val}_{\text{Nat}}(\text{True}) ; \text{val}_{\text{Nat}}(\text{False})) ; \text{val}_{\text{Nat}}(\text{False})$

Ejercicio 3 - Subtipado

Sea la extensión del Cálculo Lambda definida en la guía como cálculo μ , que permite la definición de funciones con múltiples parámetros y su aplicación parcial sobre un parámetro a elección:

$$\sigma ::= \dots \mid \{\sigma, \dots, \sigma\} \rightarrow \sigma \quad M ::= \dots \mid \mu x_1: \sigma_1, \dots, x_n: \sigma_n. M \mid M \#_i M$$

El término $\mu x_1: \sigma_1, \dots, x_n: \sigma_n. M$ sirve para construir una nueva función de n parámetros ordenados y el operador $\#_i$ sirve para aplicar el i -ésimo parámetro. Notar que si la cantidad de parámetros de una función es mayor a 1, al aplicarla se obtiene una nueva función con un parámetro menos, pero si la cantidad de parámetros es exactamente 1, al aplicarla se obtiene su valor de retorno. Notar además que el orden de los tipos de los argumentos es importante: por ejemplo, $\{\text{Nat}, \text{Nat}, \text{Bool}\} \rightarrow \text{Nat}$ y $\{\text{Bool}, \text{Nat}, \text{Nat}\} \rightarrow \text{Nat}$ no son el mismo tipo.

Las reglas de tipado son las siguientes:

$$\frac{\Gamma \cup \{x_1: \sigma_1, \dots, x_n: \sigma_n\} \triangleright M: \tau}{\Gamma \triangleright \mu x_1: \sigma_1, \dots, x_n: \sigma_n. M: \{\sigma_1, \dots, \sigma_n\} \rightarrow \tau} \text{T-}\mu \quad \frac{\Gamma \triangleright M: \{\sigma\} \rightarrow \tau \quad \Gamma \triangleright N: \sigma}{\Gamma \triangleright M \#_1 N: \tau} \text{T-APP1}$$

$$\frac{\Gamma \triangleright M: \{\sigma_1, \dots, \sigma_n\} \rightarrow \tau \quad \Gamma \triangleright N: \sigma_i \quad 1 \leq i \leq n \quad n > 1}{\Gamma \triangleright M \#_i N: \{\sigma_1, \dots, \sigma_{i+1}, \sigma_{i-1}, \dots, \sigma_n\} \rightarrow \tau} \text{T-APPN}$$

- a) Dar la(s) regla(s) de subtipado para esta extensión y justificar en términos del principio de sustitutividad.
- Pista:** no solo importan las operaciones que se pueden realizar sobre las funciones, sino también sobre los resultados que devuelven.

- b) Utilizando las reglas de tipado y subtipado, derivar el siguiente juicio de tipado:

$$\emptyset \triangleright ((\mu x: \text{Bool}, y: \text{Int}, z: \text{Float}. \text{if } x \text{ then } y \text{ else } z) \#_2 0) \#_1 \text{False}: \{\text{Nat}\} \rightarrow \text{Float}$$